Available online at www.sciencedirect.com

ScienceDirect

European Journal of Operational Research 178 (2007) 500–513

EUROPEAN JOURNAL OF OPERATIONAL RESEARCH

www.elsevier.com/locate/ejor

Decision Support

Implementing stochastic multicriteria acceptability analysis

Tommi Tervonen [a,b,c,*], Risto Lahdelma [a]

[a] University of Turku, Department of Information Technology, FIN-20520 Turku, Finland
[b] INESC—Coimbra, R. Antero de Quental 199, 3000-033 Coimbra, Portugal
[c] Faculdade de Economia da Universidade de Coimbra, Av. Dias da Silva 165, 3004-512 Coimbra, Portugal

Received 27 July 2004; accepted 29 December 2005
Available online 17 April 2006

Abstract

Stochastic multicriteria acceptability analysis (SMAA) is a family of methods for aiding multicriteria group decision making in problems with inaccurate, uncertain, or missing information. These methods are based on exploring the weight space in order to describe the preferences that make each alternative the most preferred one, or that would give a certain rank for a specific alternative. The main results of the analysis are rank acceptability indices, central weight vectors and confidence factors for different alternatives. The rank acceptability indices describe the variety of different preferences resulting in a certain rank for an alternative, the central weight vectors represent the typical preferences favouring each alternative, and the confidence factors measure whether the criteria measurements are sufficiently accurate for making an informed decision.

The computations in SMAA require the evaluation of multidimensional integrals that must in practice be computed numerically. In this paper we present efficient methods for performing the computations through Monte Carlo simulation, analyze the complexity, and assess the accuracy of the presented algorithms. We also test the efficiency of these methods empirically. Based on the tests, the implementation is fast enough to analyze typical-sized discrete problems interactively within seconds. Due to almost linear time complexity, the method is also suitable for analysing very large decision problems, for example, discrete approximations of continuous decision problems.

Keywords: Stochastic multicriteria acceptability analysis; Simulation; Multiple criteria analysis; Complexity analysis

1. Introduction

Stochastic multicriteria acceptability analysis (SMAA) methods have been developed for discrete multicriteria decision aiding (MCDA) problems, where criteria measurements are uncertain or inaccurate and where it is for some reason difficult to obtain accurate or any preference information from the decision makers (DMs) (Lahdelma and Salminen, 2001).

Usually in MCDA problems the preference information is modelled by determining importance weights for criteria. The SMAA methods are based on exploring the weight space in order to describe

* Corresponding author. Address: INESC—Coimbra, R. Antero de Quental 199, 3000-033 Coimbra, Portugal. Tel.: +351 96 529 1325; fax: +351 23 982 4692.
E-mail addresses: tommi.tervonen@it.utu.fi (T. Tervonen), risto.lahdelma@cs.utu.fi (R. Lahdelma).

0377-2217/$ - see front matter
doi:10.1016/j.ejor.2005.12.037

the preferences that would make each alternative the most preferred one, or that would give a certain rank for a specific alternative. The main results of the analysis are rank acceptability indices, central weight vectors and confidence factors for different alternatives. The rank acceptability indices describe the variety of different preferences resulting in a certain rank for an alternative, the central weight vectors represent the typical preferences favouring each alternative, and the confidence factors measure whether the criteria measurements are sufficiently accurate for making an informed decision.

In MCDA literature outside SMAA, there is a long history of methodologies that allow decision aiding under uncertain and/or imprecise information. See e.g. Dias and Clímaco (2000), Dias et al. (2002), Fishburn (1965), Hazen (1986), Kirkwood and Sarin (1985), Mousseau et al. (2000, 2003), and for more general information on this subject, see Figueira et al. (2005). Although this area has been studied for three decades, the SMAA methods are the first ones allowing both preference information and criteria measurements to be expressed as arbitrarily distributed stochastic variables. The SMAA approach has also recently been applied to extend other MCDA methods to allow using them with imprecise information (see Tervonen et al., 2005).

The SMAA methods are based on inverse weight space analysis, which has also been considered in the works of Charnetski Soland (1978) and Bana e Costa (1986). In the original SMAA method by Lahdelma et al. (1998) the weight space analysis is performed based on an additive utility or value function and stochastic criteria measurements. The SMAA-2 method (Lahdelma and Salminen, 2001) generalized the analysis to a general utility or value function, to include various kinds of preference information and to consider holistically all ranks. The SMAA-3 method (Lahdelma and Salminen, 2002) applies ELECTRE III type pseudo-criteria in the analysis. The SMAA-O method (Lahdelma et al., 2003) extends SMAA-2 for treating mixed ordinal and cardinal criteria in a comparable manner. The SMAA-A method (or Ref-SMAA method) models the preferences using reference points and achievement scalarizing functions (Lahdelma et al., 2005). Durbach (2006) has also developed a variant of the SMAA-A method using achievement functions.

SMAA methods are applicable in many real-life problem types for a number of reasons. Firstly, the inverse weight space approach is suitable for many group decision-making problems, where the DMs are unable or unwilling to provide preference information, or it is difficult to reach consensus over the preferences. In such cases the preference information can be expressed as weight intervals including preferences of all DMs, or with some other weight distribution accepted by all DMs. SMAA can then be used to compute descriptive information about the acceptability of different alternatives, and this can help the DMs to identify commonly acceptable compromise solutions. Secondly, SMAA supports a very general and flexible way to model different kinds of uncertain or inaccurate preference and criteria information through stochastic distributions. Thirdly, as demonstrated in this paper, the SMAA computations can be implemented very efficiently through numerical methods, making it possible to use the method in many different decision-making contexts, including interactive decision processes. As a consequence, SMAA methods have been successfully applied in a number of real-life decision problems in Finland. For applications of SMAA, see e.g. Hokkanen et al. (1998, 1999, 2000), Kangas et al. (2003, in press), Kangas and Kangas (2003), Lahdelma and Salminen (2006), Lahdelma et al. (2001, 2002).

In this paper we describe how the basic computations of the SMAA-2 and SMAA-O methods can be implemented efficiently through Monte Carlo simulation. We have chosen to present the computations of these two methods, because they form the basis for all other SMAA variants. In particular, we present the algorithms for computing the rank acceptability indices, central weight vectors, and confidence factors. We begin by introducing the SMAA-2 and SMAA-O methods in Section 2. In Section 3, we describe the implementation of the algorithms and discuss techniques for handling preference information. Following this, we analyze the complexity of the algorithms theoretically in Section 4. We assess the accuracy of the computations in Section 5, and present results from empirical efficiency tests in Section 6. We end this paper with conclusions in Section 7.

2. The SMAA-2 and SMAA-O methods

2.1. The basic SMAA-2 method

The SMAA-2 method (Lahdelma and Salminen, 2001) has been developed for discrete stochastic

multicriteria decision-making problems with multiple DMs. SMAA-2 applies inverse weight space analysis to describe for each alternative what kind of preferences make it the most preferred one, or place it on any particular rank. The decision problem is represented as a set of m alternatives $\{x_1, x_2, \ldots, x_m\}$ that are evaluated in terms of n criteria. The DMs' preference structure is represented by a real-valued utility or value function $u(x_i, w)$. The value function maps the different alternatives to real values by using a weight vector w to quantify DMs' subjective preferences. SMAA-2 has been developed for situations where neither criteria measurements nor weights are precisely known. Uncertain or imprecise criteria are represented by stochastic variables ξ_{ij} with joint density function $f_X(\xi)$ in the space $X \subseteq R^{m\times n}$. The DMs' unknown or partially known preferences are represented by a weight distribution with joint density function $f_W(w)$ in the feasible weight space W. Total lack of preference information is represented in 'Bayesian' spirit by a uniform weight distribution in W, that is, $f_W(w) = 1/\mathrm{vol}(W)$. The weight space can be defined according to needs, but typically, the weights are non-negative and normalized, that is; the weight space is an $n-1$-dimensional simplex in n-dimensional space:

$$W = \left\{ w \in R^n : \; w \geqslant 0 \quad \text{and} \quad \sum_{j=1}^{n} w_j = 1 \right\}. \qquad (1)$$

Fig. 1 presents the feasible weight space of a three-criterion problem as the shaded triangle with corner points $(1,0,0)$, $(0,1,0)$, and $(0,0,1)$.

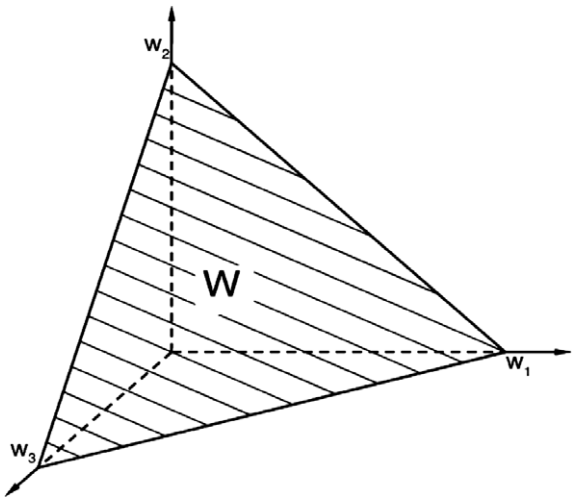

Fig. 1. Feasible weight space of a three-criterion problem.

The value function is used to map the stochastic criteria and weight distributions into value distributions $u(\xi_i, w)$. Based on the value distributions, the rank of each alternative is defined as an integer from the best rank (=1) to the worst rank (=m) by means of a ranking function

$$\mathrm{rank}(i, \xi, w) = 1 + \sum_{k=1}^{m} \rho(u(\xi_k, w) > u(\xi_i, w)), \qquad (2)$$

where $\rho(true) = 1$ and $\rho(false) = 0$. SMAA-2 is then based on analysing the stochastic sets of favourable rank weights

$$W_i^r(\xi) = \{w \in W : \mathrm{rank}(i, \xi, w) = r\}. \qquad (3)$$

Any weight $w \in W_i^r(\xi)$ results in such values for different alternatives, that alternative x_i obtains rank r.

The first descriptive measure of SMAA-2 is the *rank acceptability index* b_i^r, which measures the variety of different preferences (weights) that grant alternative x_i rank r. It is the share of all feasible weights that make the alternative acceptable for a particular rank, and it is most conveniently expressed percentage-wise. The rank acceptability index b_i^r is computed numerically as a multidimensional integral over the criteria distributions and the favourable rank weights as

$$b_i^r = \int_{\xi \in X} f_X(\xi) \int_{w \in W_i^r(\xi)} f_W(w)\, \mathrm{d}w\, \mathrm{d}\xi. \qquad (4)$$

The most acceptable (best) alternatives are those with high acceptabilities for the best ranks. Evidently, the rank acceptability indices are in the range $[0,1]$, where 0 indicates that the alternative will never obtain a given rank and 1 indicates that it will obtain the given rank always with any choice of weights.

Favourable rank weights and rank acceptability indices are illustrated in Fig. 2. The figure represents a deterministic two-criterion, three-alternative problem with linear value function. The favourable first rank weights (W_i^1) are shown in light gray, bordered by the favourable second rank weights (W_i^2) in dark gray. First and second rank acceptability indices (b_i^1, b_i^2) correspond in this figure to the distances spanned by the favourable rank weights. When the problem contains multiple criteria and alternatives, the rank acceptability indices can be better visualized by a three-dimensional column chart. Fig. 3 shows the rank acceptability indices from the Helsinki Harbour case with 13 alternatives and 11 criteria (Hokkanen et al., 1999).

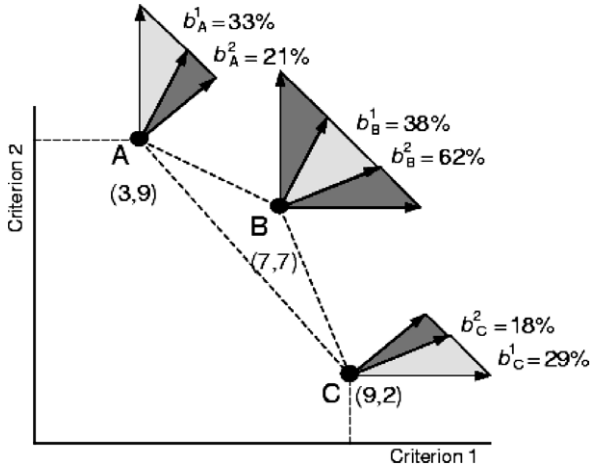

Fig. 2. First and second rank acceptabilities in a deterministic two-criterion problem with linear value function (Lahdelma and Salminen, 2001).

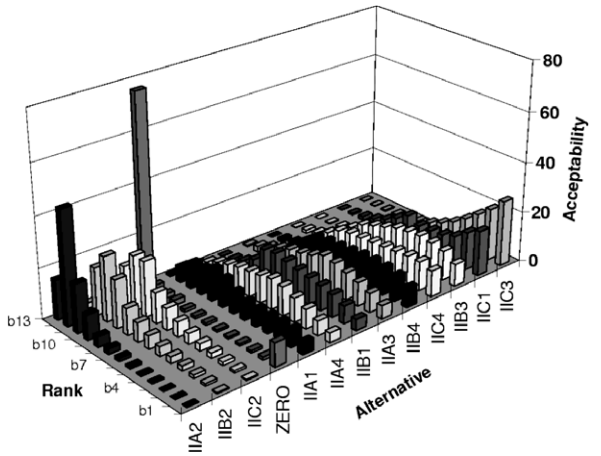

Fig. 3. Rank acceptability indices (b_i) from the Helsinki Harbour decision-making problem.

The first rank acceptability index b_i^1 is called the *acceptability index* a_i. The acceptability index is particularly interesting, because it is non-zero for stochastically efficient alternatives (alternatives that are efficient with some values for the stochastic criteria measurements) and zero for inefficient alternatives. The acceptability index not only identifies the efficient alternatives, but also measures the strength of the efficiency considering the uncertainty in criteria and DMs' preferences.

The *central weight vector* w_i^c is the expected centre of gravity (centroid) of the favourable first rank weights of an alternative. The central weight vector represents the preferences of a 'typical' DM supporting this alternative. The central weights of different alternatives can be presented to the DMs in order to help them understand how different weights correspond to different choices with the assumed preference model. The central weight vector w_i^c is computed numerically as a multidimensional integral over the criteria distributions and the favourable first rank weights using

$$w_i^c = \int_{\xi \in X} f_X(\xi) \int_{w \in W_i^1(\xi)} f_W(w) w \, \mathrm{d}w \, \mathrm{d}\xi / a_i. \tag{5}$$

Fig. 4 presents a sample chart of central weight vectors from the Helsinki Harbour decision-making problem.

The *confidence factor* p_i^c is the probability for an alternative to obtain the first rank when the central weight vector is chosen. The confidence factor is computed as a multidimensional integral over the criteria distributions using

$$p_i^c = \int_{\xi \in X : \mathrm{rank}(i,\xi,w_i^c)=1} f_X(\xi) \mathrm{d}\xi. \tag{6}$$

Confidence factors can similarly be calculated for any given weight vectors. The confidence factors measure whether the criteria measurements are accurate enough to discern the efficient alternatives.

2.2. Ordinal criteria

The SMAA-O method (Lahdelma et al., 2003) extends SMAA-2 to handle ordinal criteria measurements. In SMAA-O, the criteria may be ordinal, cardinal, or mixed. In the mixed case some of the criteria are measured on cardinal (interval) scales and others on ordinal scales. For an ordinal criterion, each alternative is measured by assigning it a *rank level*. The rank level x_{ij} is an integer from the best rank level 1 to the worst rank level m_j. Observe that multiple alternatives may obtain the same rank level, in which case $m_j < m$. The idea in SMAA-O is to map the ordinal criteria measurements into cardinal scales before they are used in the computations. The mapping is implemented by a function $g_j(\cdot)$ that preserves the ordinal information:

$$x_{ij} \succ x_{kj} \Longleftrightarrow g_j(x_{ij}) > g_j(x_{kj}) \quad \forall i,k \in \{1,\ldots,m\}. \tag{7}$$

Without loss of generality we can assume that the mapping is scaled to interval $[0,1]$. Otherwise the

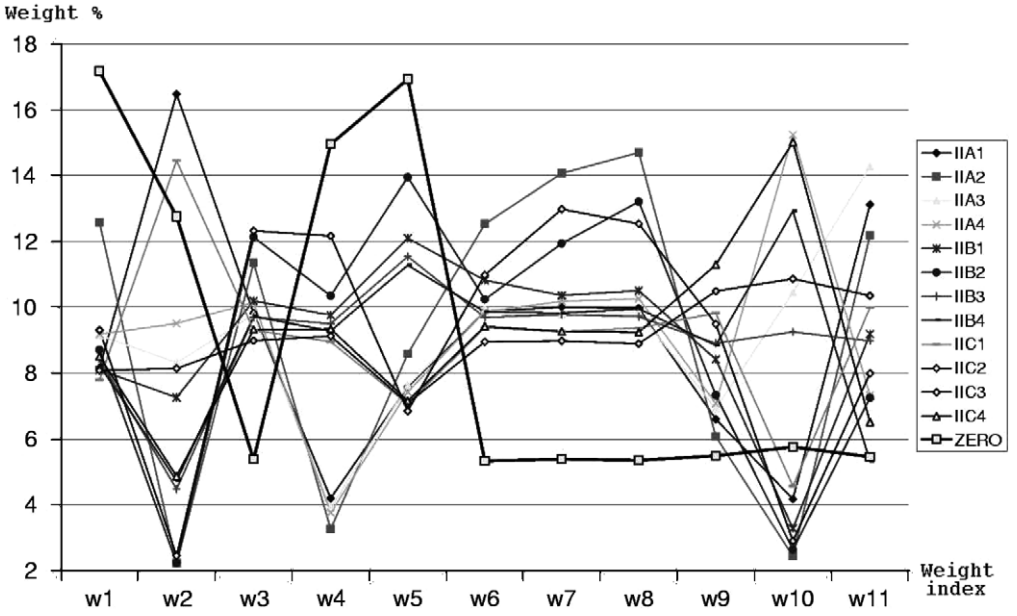

Fig. 4. Central weight vectors from the Helsinki Harbour decision-making problem.

shape of the mapping is unknown. SMAA-O simulates numerically all such mappings that preserve the ordinal criteria information.

2.3. Preference information

There are several different ways to handle partial preference information in SMAA methods. In this paper we focus on two ways that are applicable when the value function is additive (Lahdelma and Salminen, 2001):

- interval constraints for weights, and
- complete ranking of the criteria.

The preference information might also be mixed: there might be exact numerical values for some weights, ranking for a set of weights, and interval constraints for some weights. Mixed preference information is not considered in this paper.

Interval constraints for weights are given in the form

$$0 \leqslant w_j^{\min} \leqslant w_j \leqslant w_j^{\max} \leqslant 1, \quad \text{where } j \in \{1, \ldots, n\}. \tag{8}$$

The intervals $[w^{\min}, w^{\max}]$ can be defined so that they contain the preferences of the DMs (and other interest groups). The DMs can express their preferences either as precise weights or as weight intervals. The weight space analysis of SMAA is then performed in the restricted weight space

$$W' = \{w \in W | w_j^{\min} \leqslant w_j \leqslant w_j^{\max},\ j = 1, \ldots, n\}. \tag{9}$$

This means that the uniform weight distribution $f_W(w)$ is redefined as

$$f_W(w) = \begin{cases} 1/\mathrm{vol}(W') & \text{if } w \in W', \\ 0 & \text{otherwise.} \end{cases} \tag{10}$$

Fig. 5 illustrates the restricted feasible weight space of a three-criterion problem with lower and upper bounds for w_1.

Complete ranking of the criteria is expressed as a sequence of inequality constraints for the weights

$$w_{j_1} \geqslant w_{j_2} \geqslant \cdots \geqslant w_{j_n}. \tag{11}$$

Such a ranking can be obtained by asking the DMs to identify the most important, second most important, etc. criterion. When judging mutual importance of the criteria, the DMs should consider the difference between the best and worst value for each criterion. If the DMs consider two criteria equally important, this can be represented by an equality constraint between those criteria in (11). Fig. 6 illustrates the feasible weight space for a three-criterion problem with the ranking $w_1 \geqslant w_2 \geqslant w_3$.

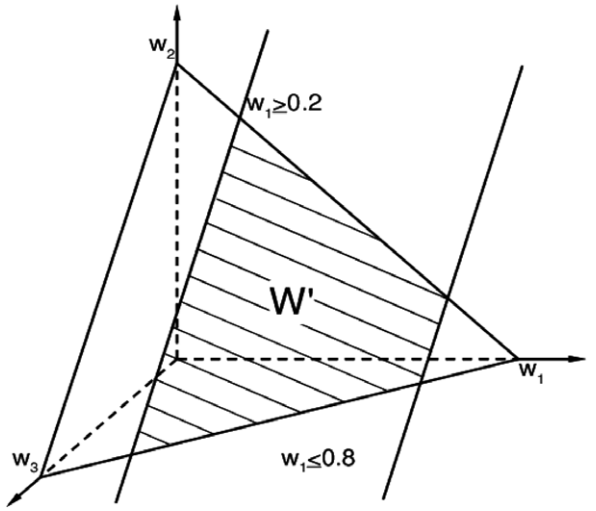

Fig. 5. Feasible weight space of a three-criterion problem with lower and upper bounds for w_1.

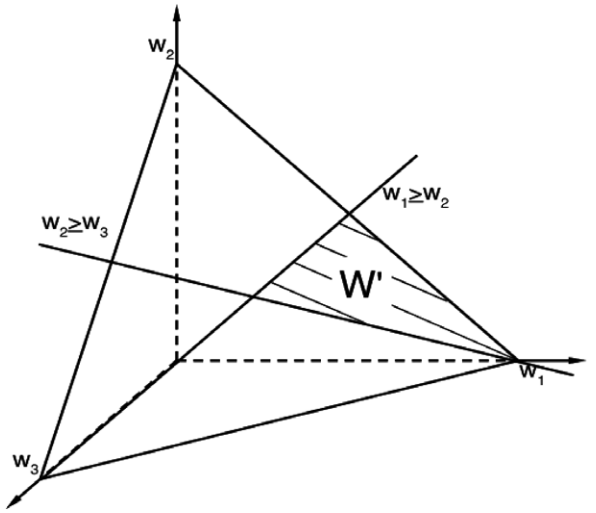

Fig. 6. Feasible weight space of a three-criterion problem with ranking of the criteria.

3. Description of the SMAA algorithm

The multidimensional integrals (4)–(6) of SMAA computations are in practice impossible to compute analytically, because the distributions f_X and f_W vary according to the application and can be arbitrarily complex. Straightforward integration techniques based on discretizing the distributions with respect to each dimension are infeasible, because the integrals have a very high dimension, and the required effort depends exponentially on the number of dimensions. For example, in a problem with eight criteria and 10 alternatives, the dimension of the integral for computing rank acceptability indices is 88, because in (4) the outer integration is through the eight-dimensional criteria space, and the inner one through the space of all criteria measurements for all alternatives ($8 \times 10 = 80$ dimensions). However, due to the nature of the problem, we do not need an answer with very high precision. Monte Carlo simulation is a well-established method for computing approximative values for high-dimensional integrals. In Monte Carlo simulation the required number of iterations is inversely proportional to the square of the desired accuracy, but does not significantly depend on the dimensionality of the problem (Fishman, 1996). Thus, Monte Carlo simulation can be used to obtain a precision of a few decimal places with moderate effort.

The algorithm is described in four parts. We first describe the method for generating a criterion matrix with cardinal and ordinal criteria. Secondly, we describe the applied weight generation technique and how preference information is handled. Before we describe the actual SMAA algorithm, we observe that the confidence factors (6) depend both on the central weight vectors and on the acceptability indices. As a consequence, the algorithm must consist of two phases. *Phase 1* consists of computation of the rank acceptability indices and the central weight vectors. The confidence factors are computed in *Phase 2*.

The following symbols are used in Algorithms 1–4:

- h_i^j number of times alternative i is evaluated into rank j in Monte Carlo simulations of Phase 1 (hits for rank j of alternative i)
- K_w number of iterations in Phase 1 (rank acceptability index and central weight vector computation)
- K_c number of iterations in Phase 2 (confidence factor computation)
- m_j number of rank levels for ordinal criterion j
- $r = [r_1, \ldots, r_m]$ vector of ranks of the alternatives
- $t = [t_1, \ldots, t_m]$ vector of value function values of the alternatives

The algorithms also use the following functions and subroutines:

- $\mathrm{RAND}_{U[0,1]}()$ function returning a uniformly distributed random number from the interval $[0, 1]$
- $\mathrm{RAND}_X()$ function returning a random criterion matrix from criteria distribution f_X

$\mathrm{RAND}_W()$ function returning a random weight vector from weight distribution f_W

$\mathrm{RANK}(t)$ function returning a vector of ranks corresponding to the vector of value function values t

$\mathrm{SORT}_{\mathrm{asc}}(s)$ subroutine sorting the components of vector s into ascending order

$\mathrm{SORT}_{\mathrm{desc}}(s)$ subroutine sorting the components of vector s into descending order

3.1. Generation of the criteria measurement matrix

The $\mathrm{RAND}_X()$ function generates a random criterion matrix of size $m \times n$ from the given criteria distribution. Each row of the matrix contains criteria measurements of a certain alternative. Cardinal criteria measurements follow a joint distribution, or independent distributions. We do not consider joint distribution for cardinal criteria in this paper. Refer to Lahdelma et al. (2004). Independent criteria measurements are generated separately from their corresponding distributions. Their distributions may have an arbitrary shape (e.g.uniform, normal, ...).

If some of the criteria are measured on ordinal scales, then the ordinal to cardinal mapping must be simulated for those criteria each time a new criterion matrix is created. The ordinal to cardinal mapping is simulated using the following method: first, $m_j - 2$ uniformly distributed random numbers from the interval $]0,1[$ are generated and sorted into descending order (m_j is the number of rank levels). Then, 1 is inserted as the first number and 0 as the last number. The simulated cardinal value for rank level j is then the jth of these numbers. Thus the simulated cardinal value for the best rank level is 1, and the simulated cardinal value for the worst rank level is 0. The simulated cardinal values for other rank levels should be unique and in the interval $]0,1[$. Because the majority of pseudo-random number generators will not produce duplicate floating point values except after a very long sequence, it is in practice unnecessary to have any special treatment for duplicate values. The procedure for generating the simulated cardinal values is defined as pseudo-code in Algorithm 1. Complexity of this procedure is due to sorting $O(m\log(m))$. The procedure must be executed once for each ordinal criterion when a new criterion matrix is generated. Fig. 7 illustrates a possible mapping with 11 rank levels generated by this procedure.

Algorithm 1. Generation of m_j simulated cardinal values $(q_1, \ldots, q_{m_j})$.

Output: q
1: **for** $j \leftarrow 2$ **to** $m_j - 1$ **do**
2: $\quad q_j \leftarrow \mathrm{RAND}_{U[0,1]}()$
3: **end for**
4: $\mathrm{SORT}_{\mathrm{desc}}(q)$
5: $q_1 \leftarrow 1$
6: $q_{m_j} \leftarrow 0$

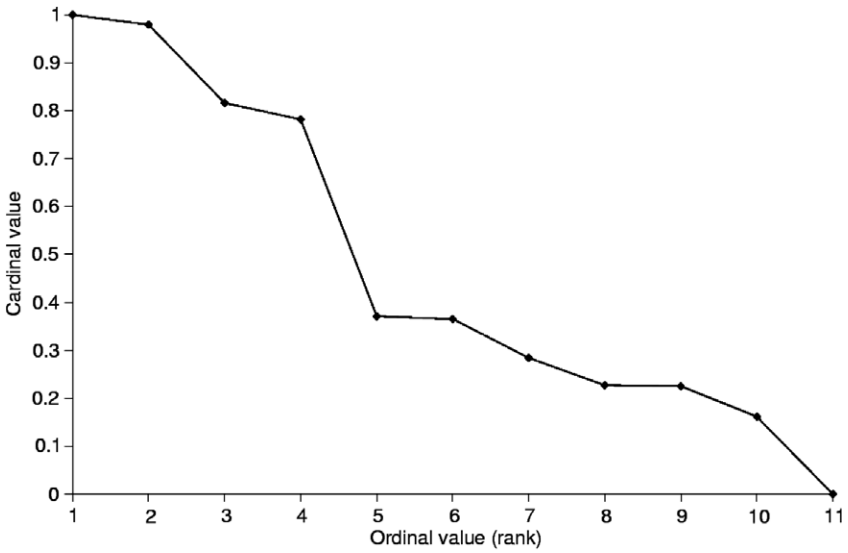

Fig. 7. A sample ordinal to cardinal mapping with $m_j = 11$ rank levels.

3.2. Generation of weights and handling preference information

The $\mathrm{RAND}_W()$ function generates the weights from the given weight distribution. We describe weight generation corresponding to three different types of preference information:

- absent preference information,
- interval constraints for weights, and
- complete ranking of the criteria.

In case of absent preference information the weights are generated from a uniform distribution in the normalized weight space (1). Because the weights must sum to unity, the n weights w_j are generated according to the following method: first $n-1$ independent random numbers are generated from the uniform distribution in interval $[0,1]$, and sorted into ascending order $(q_1, q_2, \ldots, q_{n-1})$. After that, 1 is inserted as the last number ($q_n = 1$) and 0 as the first number ($q_0 = 0$). Uniformly distributed normalized weights are then obtained as intervals between the consecutive numbers ($w_j = q_j - q_{j-1}$) (David, 1970). The procedure for generating the n uniformly distributed normalized weights is defined in Algorithm 2. Complexity of this procedure is $O(n\log(n))$ due to sorting.

When preference information is available, the weight generation process must be modified a little. Upper and lower bounds for weights (and in principle also more complex weight constraints) can be implemented by the rejection technique. After a vector of uniformly distributed normalized weights has been generated, the weights are tested against their bounds. If any of the constraints is not satisfied, the entire set is rejected and the weight generation is repeated. A problem with the rejection technique is that it may cause a very large share of the weights to be rejected and a very small share of them to be accepted. A small acceptance rate not only slows down the computation, but may also cause problems with the quality of the generated pseudo-random numbers that pass the rejection test. However, upper and lower bounds affect the acceptance rate differently. As can be seen from Fig. 5, upper bounds cut off the tip of the simplex, but lower bounds cut off the base. In a high-dimensional weight space, the volume of the tip is very small, but the volume of the base is large in relation to the entire weight space.

To estimate how large a share of the weight vectors need to be rejected due to upper bounds, we assume that all weights have a common upper bound $w^{\max}$. Then the probability for the largest of the generated weights to exceed the upper bound is

$$\begin{aligned} P[\max\{w_j\} > w^{\max}] &= n(1-w^{\max})^{n-1} - \binom{n}{2}(1-2w^{\max})^{n-1} \\ &\quad + \cdots (-1)^{k-1}\binom{n}{k}(1-kw^{\max})^{n-1}\cdots, \end{aligned} \tag{12}$$

where the series continues as long as $1 - kw^{\max} > 0$ (David, 1970). For example, if there are $n = 5$ weights with upper bound $w^{\max} = 0.4$, the rejection percentage is 63.2%. If we are applying both lower and upper bounds, it might be the case, that the lower bounds render some upper bounds redundant. Consider a three-criterion case with a lower bound of 0.3 for all weights. Then the maximum value that any weight may obtain is $1 - 0.3 - 0.3 = 0.4$. Therefore all upper bound weight constraints of 0.4 or higher are redundant.

The rejection technique can be very inefficient for weights with lower bounds in high-dimensional problems. Lower bounds can be treated efficiently by using a simple transformation technique. With lower bounds the feasible weight space becomes

$$W' = \left\{ w \in R^n | w_j \geqslant w_j^{\min} \quad \text{and} \quad \sum_{j=1}^{n} w_j = 1 \right\}, \tag{13}$$

which has the same simplex shape as the original weight space W, but is smaller. By substituting $w'_j = w_j - w_j^{\min}$ the restricted weight space becomes

$$W' = \left\{ w' \in R^n | w'_j \geqslant 0 \quad \text{and} \quad \sum_{j=1}^{n} w'_j = 1 - C \right\}, \quad \text{where } C = \sum_{j=1}^{n} w_j^{\min}. \tag{14}$$

The shifted weights w'_j can now be generated by a modification of Algorithm 2 where the weights are generated to sum to $1 - C$ instead of 1. Then the lower bounded weights w_j are obtained by substituting back $w_j = w'_j + w_j^{\min}$. Lower bounds do therefore not increase the complexity of weight generation.

Preference information presented in form of a complete ranking of the criteria is handled using a similar technique that was used for simulating the

ordinal to cardinal mapping in the previous section. First we generate a set of weights from a uniform distribution in the normalized weight space as in the case of absent preference information. Then we sort the weights into a consistent order according to the ranking of the criteria. This does not increase the complexity of weight generation.

Algorithm 2. Generation of n uniformly distributed random weights from the interval $[0,1]$ $(w_1,\ldots,w_n)$ which sum to unity.

Output: w
1: **for** $j \leftarrow 1$ **to** $n-1$ **do**
2: $\quad q_j \leftarrow \mathrm{RAND}_{U[0,1]}()$
3: **end for**
4: $\mathrm{SORT}_{\mathrm{asc}}(q)$
5: $q_0 \leftarrow 0$
6: $q_n \leftarrow 1$
7: **for** $j \leftarrow 1$ **to** n **do**
8: $\quad w_j \leftarrow q_j - q_{j-1}$
9: **end for**

3.3. Phase 1. Computation of b_i^r and w_i^c

To compute the rank acceptability indices b_i^r and the central weight vectors w_i^c for each alternative i, we must integrate over the criteria and weight distributions. Straightforward computation of the rank acceptability indices (4) would require executing Monte Carlo simulation $m \cdot n$ times, once for each index. Similarly, computing the central weight vector (5) for each alternative would require m executions. We can speed up the computation remarkably by observing that all rank acceptability indices and central weight vectors can be computed in a single simulation run. To do this, we generate during each iteration a random criterion matrix and a random weight vector from their corresponding distributions. Then we compute statistics on the ranks that different alternatives obtained and update the central weight vector of the most preferred alternative. Phase 1 is described as pseudocode in Algorithm 3.

Algorithm 3 uses the function RANK(t). RANK(t) returns a vector of ranks for alternatives based on their values in vector t. For example, if $t = [0, 0.5, 0.2]$, the resulting rank vector is $[3,1,2]$. This function is implemented efficiently by sorting the alternatives into descending order by their values and then assigning consecutive ranks from 1 to m to the sorted alternatives. Sometimes two or more alternatives may have the same values, and they should thus be assigned the same rank. Rank assignment should be implemented to handle such cases properly. However, shared ranks will be extremely rare when the criteria measurements are stochastic and independent. Complexity of this procedure is due to sorting $O(m\log(m))$.

Observe in Algorithm 3 that the central weight vector (w_i^c) is defined only when the acceptability index is non-zero, or, equivalently, when hits for the first rank (h_i^1) is greater than zero.

Algorithm 3. Monte Carlo simulation to compute the central weight vectors (w_i^c's) and the acceptability indices (b_i^r's).

Output: w_i^c's, b_i^r's
1: // Initialization of w_i^c and hit count
2: **for** $i \leftarrow 1$ **to** m **do**
3: $\quad w_i^c \leftarrow 0$
4: $\quad$ **for** $j \leftarrow 1$ **to** m **do**
5: $\quad\quad h_i^j \leftarrow 0$
6: $\quad$ **end for**
7: **end for**
8: // Main loop
9: **for** $k \leftarrow 1$ **to** K_w **do**
10: $\quad w \leftarrow \mathrm{RAND}_W()$
11: $\quad x \leftarrow \mathrm{RAND}_X()$
12: $\quad$ **for** $i \leftarrow 1$ **to** m **do**
13: $\quad\quad t_i \leftarrow u(x_i, w)$
14: $\quad$ **end for**
15: $\quad r \leftarrow \mathrm{RANK}(t)$
16: $\quad$ **for** $i \leftarrow 1$ **to** m **do**
17: $\quad\quad h_i^{r_i} \leftarrow h_i^{r_i} + 1$
18: $\quad\quad$ **if** $r_i = 1$ **then**
19: $\quad\quad\quad w_i^c \leftarrow w_i^c + w$
20: $\quad\quad$ **end if**
21: $\quad$ **end for**
22: **end for**
23: // Computation of w_i^c and b_i^r
24: **for** $i \leftarrow 1$ **to** m **do**
25: $\quad$ **if** $h_i^1 > 0$ **then**
26: $\quad\quad w_i^c \leftarrow w_i^c / h_i^1$
27: $\quad$ **end if**
28: $\quad$ **for** $j \leftarrow 1$ **to** m **do**
29: $\quad\quad b_i^j \leftarrow h_i^j / K_w$
30: $\quad$ **end for**
31: **end for**

3.4. Phase 2. Computation of p_i^c

To compute the confidence factors p_i^c from (6) we must integrate over the criteria distribution with respect to the different central weight vectors. Naive implementation of the computation would require repeating the simulation m times, once for each alternative. Again, we can device a way to compute all integrals simultaneously. To do this, we first generate during each iteration a random criterion matrix from the appropriate criteria distribution. After that, we evaluate for each alternative whether that alternative is the most preferred one using its central weight vector and the random criterion matrix. This technique decreases the number of generated criterion matrices by a factor of m. However, to evaluate if the alternative is the most preferred one, we still need to evaluate the value function a maximum of m times in an inner loop. We shall see later a surprising result on the expected complexity of this algorithm. The algorithm for Phase 2 is presented as pseudo-code in Algorithm 4.

Algorithm 4. Monte Carlo simulation to compute the confidence factors (p_i^c's).

Output: p_i^c's
1: **for** $i \leftarrow 1$ **to** m **do**
2: $p_i^c \leftarrow 0$
3: **end for**
4: **for** $j \leftarrow 1$ **to** K_c **do**
5: $x \leftarrow \mathrm{RAND}_X()$
6: **for** $i \leftarrow 1$ **to** m **do**
7: $t \leftarrow u(x_i, w_i^c)$
8: **for all** $k \in \{1,\ldots,m\}\backslash\{i\}$ **do**
9: **if** $u(x_k, w_i^c) > t$ **then**
10: **goto** worse
11: **end if**
12: **end for**
13: $p_i^c \leftarrow p_i^c + 1$
14: worse:
15: **end for**
16: **end for**
17: **for** $i \leftarrow 1$ **to** m **do**
18: $p_i^c \leftarrow p_i^c / K_c$
19: **end for**

4. Complexity of the SMAA algorithm

If all of the criteria are measured on cardinal scales, the complexity of the algorithm for Phase 1 (Algorithm 3) is $O(K_w \cdot \phi_W + K_w \cdot \phi_X + K_w \cdot m \cdot n + K_w \cdot m\log(m) + m^2)$, where ϕ_W is the complexity of generating a weight vector from the weight distribution and ϕ_X is the complexity of generating a criterion matrix from the criteria distribution. In many applications the weights are generated from a uniform distribution following the method described before (Algorithm 2). In practice the number of iterations $K_w \gg m$, because K_w is fairly large (10^4–10^6) to obtain sufficient accuracy and m is fairly small. With these assumptions the complexity can be written as $O(K_w \cdot (n\log(n) + m \cdot n + \phi_X + m\log(m)))$. If criteria measurements are independent, the complexity is $O(K_w \cdot (n\ \log(n) + m \cdot n + m\log(m)))$. In many practical decision-making problems the term $m \cdot n$ dominates, and the complexity can thus be written as $O(K_w \cdot m \cdot n)$. If some of the criteria are measured on ordinal scales, the ordinal to cardinal mapping is required for those criteria, and in that case the total complexity of Algorithm 2 is $O(K_w \cdot n \cdot m\log(m))$.

The complexity of the algorithm for Phase 2 (Algorithm 4) is $O(K_c \cdot (\phi_X + m^2 \cdot n))$, if all of the criteria are measured on cardinal scales. During each iteration, the algorithm first generates a criterion matrix for all alternatives, and then uses these when computing the values with different central weight vectors. This decreases the number of criterion matrices generated during Phase 2 of the algorithm from $K_c \cdot m$ to K_c and affects the running time remarkably. If the criteria measurements are independent, the complexity of the algorithm for Phase 2 can be written as $O(K_c \cdot m^2 \cdot n)$.

The algorithm for Phase 2 which has squared worst-case complexity with respect to m, has in fact quite low typical-case complexity. The squared complexity is a consequence of the inner loop comparing values of the alternatives. However, that loop is almost never executed completely. Normally, when the criteria measurements are independent stochastic variables, the values of the alternatives will be distinct. If we assume that during each Monte Carlo iteration one of the alternatives will have the best value with respect to its central weight vector, one the second best, etc., and if the alternatives are compared in a random order, then the

expected number of value function evaluations during each Monte Carlo iteration is

$$(1+(m-1))+\left(1+\frac{m}{2}\right)+\left(1+\frac{m}{3}\right)+\cdots+\left(1+\frac{m}{m}\right) = m-1+m\sum_{i=1}^{m}\frac{1}{i} = m-1+m\cdot H_m. \quad (15)$$

In this formula, the mth subsum of the harmonic series is known as the harmonic number H_m. H_m grows very slowly, so the typical-case complexity of the algorithm is not squared with respect to m, but rather close to linear. The typical-case complexity can be written as $O(K_c \cdot H_m \cdot m \cdot n)$. Again, if criteria are measured on ordinal scales, the complexity of the ordinal to cardinal mapping procedure increases the total typical-case complexity of Algorithm 4 to $O(K_c \cdot H_m \cdot n \cdot m\log(m))$.

5. Accuracy of the SMAA computations

The accuracy of the results can be calculated by considering the Monte Carlo simulations as point estimators for b_i^r and p_i^c. By the central limit theorem we can conclude that b_i^r and p_i^c are normally distributed, if the numbers of iterations (K_w, K_c) are large enough (>25) (Milton and Arnold, 1995). In practical SMAA computations the number of iterations is typically 10^4–10^6.

If we want to achieve accuracy d_b with 95% confidence for b_i^r, we need the following number of Monte Carlo iterations K_w (Milton and Arnold, 1995):

$$K_w = \frac{1.96^2}{4d_b^2}. \quad (16)$$

For example, if we want to achieve error limits of 0.01 for b_i^r, that can be accomplished with 95% confidence by performing $K_w = 9604$ Monte Carlo iterations.

The accuracy of p_i^c depends on the accuracy of the central weight vectors and the criteria distribution in a complex manner. In theory, an arbitrarily small error in a central weight vector may cause an arbitrarily large error in a confidence factor. If we disregard this error source for the confidence factors, then the same accuracy analysis applies for the confidence factors as for the rank acceptability indices, that is, $K_c = K_w$ yields the same precision for the confidence factors as for the rank acceptability indices.

The accuracy of w_i^c does not depend on the total number of Monte Carlo iterations, but rather on the number of iterations that contribute to the computation of that central weight vector. To achieve an accuracy of d_w with 95% confidence for w_i^c, the required number of iterations is

$$K_c = \frac{1.96^2}{a_i \cdot 4d_w^2}. \quad (17)$$

Thus, alternatives with small acceptability indices require more iterations to compute their central weight vectors with a given accuracy. In practice, we are normally not interested in central weight vectors for alternatives with extremely low acceptability indices.

6. Empirical tests

We have performed empirical tests to measure the running time of the algorithm separately for Phases 1 and 2. The tests were performed on a GNU/Linux personal computer with one 2.6 GHz Pentium-4 processor and no significant extra load during the tests.

Our test problems include all combinations of the number of alternatives m and number of criteria n, where

$$m \in \{4, 6, 8, 10, 15, 25, 50, 100, 150, 200\} \quad \text{and}$$
$$n \in \{4, 8, 16, 32\}.$$

For each problem size we generated six sample problems: three with uniformly distributed cardinal criteria measurements (SMAA-2), and three with ordinal criteria measurements (SMAA-O). The cardinal criteria measurements were uniformly distributed in the intervals $[x_{ij} - 0.2, x_{ij} + 0.2]$ where the mean x_{ij} was chosen randomly from the interval $[0, 1]$. The ordinal measurements had distinct random rank levels for all criteria. For both SMAA-2 and SMAA-O, one problem contained no preference information (NOP), another included complete ranking of the criteria (ORDP), and the third contained preference information in form of weight intervals $[1/(n*2), 0.5]$ for all criteria (INTP). Thus, we have a total of $10 \cdot 4 \cdot 6 = 240$ test problems.

For each test run we used 10 000 Monte Carlo iterations ($K_w = K_c = 10\,000$). Based on Eq. (16) this yields an accuracy of $d < 0.01$ for the rank acceptability indices, which is sufficient in many real-life applications.

We executed the SMAA algorithm 20 times for all test problems. After that, we calculated the mean of the running time for each model. Total running times are presented in Fig. 8. The times for test runs with weight intervals are not shown in the figure, because including weight intervals had no observable impact on the running time when compared with no preference information. From Fig. 8 we can see that when the problem size grows, SMAA-O is clearly slower than SMAA-2. Still, both methods are fast enough to be used in practical decision-making situations. Inclusion of ordinal preference information does not significantly increase the running time of SMAA-O, and imposes only a minor increase to the running time of SMAA-2.

To analyze the complexity of Phases 1 and 2 in more detail, we have computed the running times in milliseconds divided by the product of number of alternatives and criteria. We have plotted these times as stacked columns for SMAA-2 in Fig. 9 and for SMAA-O in Fig. 10. Note that some combinations of (m,n) result in duplicate labels on the x-axis. From these figures it can be seen, that the

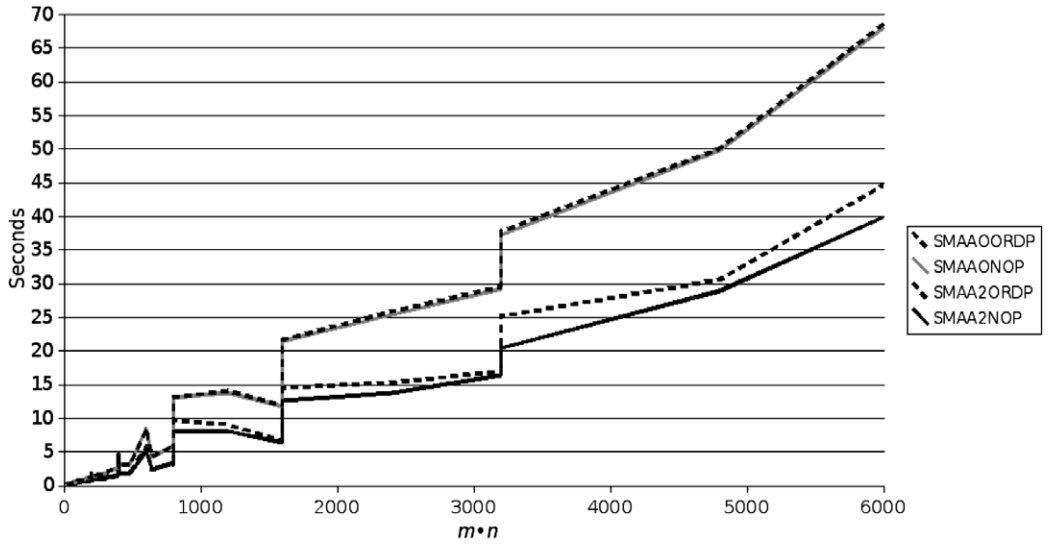

Fig. 8. Total running times of tests.

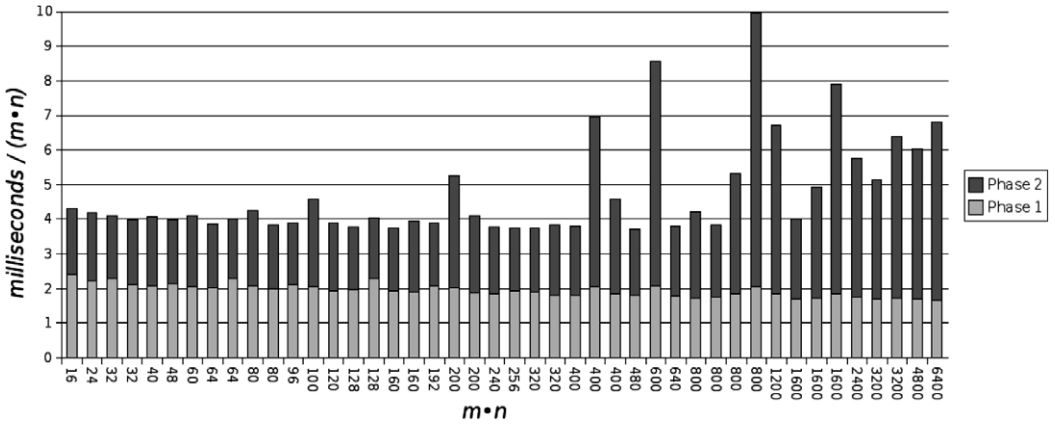

Fig. 9. Ratio of running time of SMAA-2 (in milliseconds) and product of number of alternatives and criteria.

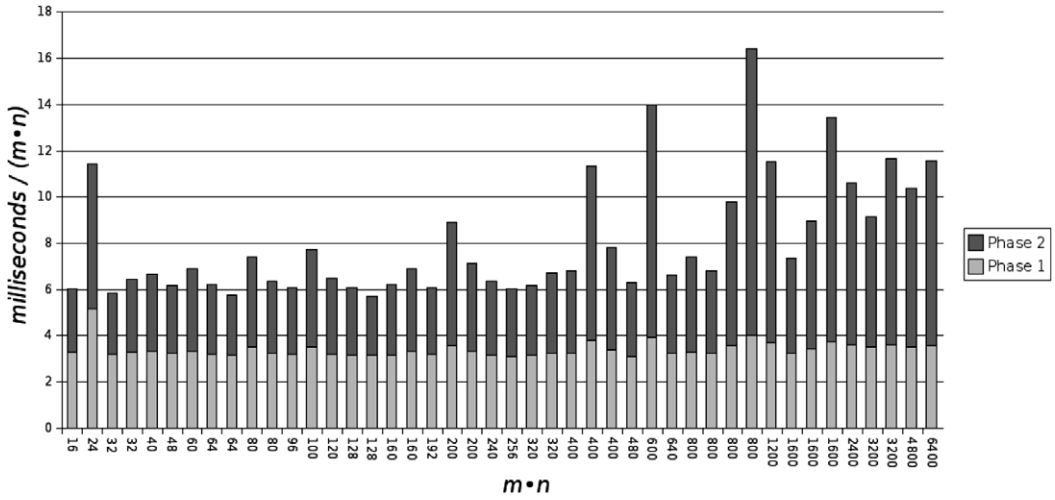

Fig. 10. Ratio of running time of SMAA-O (in milliseconds) and product of number of alternatives and criteria.

Phase 1 indeed has linear running time in respect to the number of criteria and alternatives. The running time for Phase 2 grows a little faster than the factor H_m would indicate. This is due to the fact that our implementation does not randomize the order in which the alternatives are compared in the innermost loop.

The empirical test results show that SMAA methods are applicable in MCDA problems with a large number of criteria and alternatives. In a typical MCDA ranking or choosing problem, there are under 20 alternatives and criteria. In this case, the execution of the algorithm with a personal computer takes only a few seconds. It should also be noted, that the total execution time grows almost linearly with respect to the number of alternatives and criteria. Therefore, these algorithms can be used also with very large decision-making problems (for example, over 100 criteria and over 1000 alternatives).

7. Conclusions

Stochastic multicriteria acceptability analysis is a family of methods for aiding multicriteria group decision making in problems with inaccurate, uncertain, or missing information. The multidimensional integrals which form core of the SMAA computations are in practice impossible to compute analytically. We have demonstrated that the computations can be implemented efficiently with sufficient accuracy using Monte Carlo simulation techniques. With cardinal criteria, the computation time is nearly proportional to $n \cdot m$. With ordinal criteria, the computation time is nearly proportional to $n \cdot m \cdot \log(m)$.

In a group decision-making process, it is common that new preference information is received and old information is adjusted as the process evolves. When new information is added to the model, the SMAA computations must be repeated. The empirical efficiency tests of the presented implementations show that the required time for computing a typical decision-making problem with 10 alternatives and eight criteria with a personal computer is less than a second. Thus, the effect of modified preference information on the results can be investigated interactively by the decision makers.

Some decision-making problems are continuous by nature, and the number of alternatives is thus in principle infinite. One approach to solve such problems is to form a large number of discrete decision alternatives and to evaluate them using discrete decision support methods. SMAA methods can help in such processes to filter out alternatives that are inefficient or otherwise inferior. The results in this paper show that SMAA methods are fast enough also to be used in such, fairly large problems.

Acknowledgements

The work of Tommi Tervonen was partially supported by the MONET research project (POCTI/GES/37707/2001) and grants from Turun Yliopistosäätiö and the Finnish Cultural Foundation.

References

Bana e Costa, C.A., 1986. A multicriteria decision aid methodology to deal with conflicting situations on the weights. European Journal of Operational Research 26, 22–34.

Charnetski, J., Soland, R., 1978. Multiple-attribute decision making with partial information: The comparative hypervolume criterion. Naval Research Logistics Quarterly 25, 279–288.

David, H.A., 1970. Order Statistics. Wiley, New York.

Dias, L., Clímaco, J., 2000. ELECTRE TRI for groups with imprecise information on parameter values. Group Decision and Negotiation 9 (5), 355–377.

Dias, L., Mousseau, V., Figueira, J., Clímaco, J., 2002. An aggregation/disaggregation approach to obtain robust conclusions with ELECTRE TRI. European Journal of Operational Research 138, 332–348.

Durbach, I., 2006. A simulation-based test of stochastic multicriteria acceptability analysis using achievement functions. European Journal of Operational Research 170 (3), 923–934.

Figueira, J., Greco, S., Ehrgott, M. (Eds.), 2005. Multiple Criteria Decision Analysis: State of the Art Surveys. Springer Science+Business Media, Inc., New York.

Fishburn, P., 1965. Analysis of decisions with incomplete knowledge of probabilities. Operations Research 13, 217–237.

Fishman, G., 1996. Monte Carlo: Concepts, Algorithms, and Applications. Springer-Verlag, New York.

Hazen, G., 1986. Partial information, dominance, and potential optimality in multiattribute utility theory. Operations Research 34, 297–310.

Hokkanen, J., Lahdelma, R., Miettinen, K., Salminen, P., 1998. Determining the implementation order of a general plan by using a multicriteria method. Journal of Multi-Criteria Decision Analysis 7 (5), 273–284.

Hokkanen, J., Lahdelma, R., Salminen, P., 1999. A multiple criteria decision model for analyzing and choosing among different development patterns for the Helsinki cargo harbor. Socio-Economic Planning Sciences 33, 1–23.

Hokkanen, J., Lahdelma, R., Salminen, P., 2000. Multicriteria decision support in a technology competition for cleaning polluted soil in Helsinki. Journal of Environmental Management 60, 339–348.

Kangas, A., Kangas, J., Lahdelma, R., Salminen, P., in press. Using SMAA-2 method with dependent uncertainties for strategic forest planning. Forest Policy and Economics.

Kangas, J., Kangas, A., 2003. Multicriteria approval and SMAA-O method in natural resources decision analysis with both ordinal and cardinal criteria. Journal of Multi-Criteria Decision Analysis 12, 3–15.

Kangas, J., Hokkanen, J., Kangas, A., Lahdelma, R., Salminen, P., 2003. Applying stochastic multicriteria acceptability analysis to forest ecosystem management with both cardinal and ordinal criteria. Forest Science 49 (6), 928–937.

Kirkwood, C., Sarin, R., 1985. Ranking with partial information: A method and an application. Operations Research 33, 38–48.

Lahdelma, R., Salminen, P., 2001. SMAA-2: Stochastic multicriteria acceptability analysis for group decision making. Operations Research 49 (3), 444–454.

Lahdelma, R., Salminen, P., 2002. Pseudo-criteria versus linear utility function in stochastic multi-criteria acceptability analysis. European Journal of Operational Research 14, 454–469.

Lahdelma, R., Salminen, P., 2006. Classifying efficient alternatives in SMAA using cross confidence factors. European Journal of Operational Research 170 (1), 228–240.

Lahdelma, R., Hokkanen, J., Salminen, P., 1998. SMAA—Stochastic multiobjective acceptability analysis. European Journal of Operational Research 106, 137–143.

Lahdelma, R., Salminen, P., Simonen, A., Hokkanen, J., 2001. Choosing a reparation method for a landfill using the SMAA-O multicriteria method. In: Multiple Criteria Decision Making in the New Millenium. Lecture Notes in Economics and Mathematical Systems, vol. 507. Springer-Verlag, Berlin, pp. 380–389.

Lahdelma, R., Salminen, P., Hokkanen, J., 2002. Locating a waste treatment facility by using stochastic multicriteria acceptability analysis with ordinal criteria. European Journal of Operational Research 142, 345–356.

Lahdelma, R., Miettinen, K., Salminen, P., 2003. Ordinal criteria in stochastic multicriteria acceptability analysis (SMAA). European Journal of Operational Research 147, 117–127.

Lahdelma, R., Makkonen, S., Salminen, P., 2004. Treating dependent uncertainties in multi-criteria decision problems. In: Rao, M.R., Puri, M.C. (Eds.), Operational Research and Its Applications: Recent Trends, vol. II, New Delhi, pp. 427–434.

Lahdelma, R., Miettinen, K., Salminen, P., 2005. Reference point approach for multiple decision makers. European Journal of Operational Research 164 (3), 785–791.

Milton, J.S., Arnold, J.C., 1995. Introduction to Probability and Statistics. Probability and Statistics, 3rd ed. McGraw-Hill International editions.

Mousseau, V., Słowiński, R., Zielniewicz, P., 2000. A user-oriented implementation of the ELECTRE-TRI method integrating preference elicitation support. Computers & Operations Research 27 (7–8), 757–777.

Mousseau, V., Figueira, J., Dias, L., Gomes da Silva, C., Clímaco, J., 2003. Resolving inconsistencies among constraints on the parameters of an MCDA model. European Journal of Operational Research 147, 72–93.

Tervonen, T., Almeida-Dias, J., Figueira, J., Lahdelma, R., Salminen, P., 2005. SMAA-TRI: A parameter stability analysis method for ELECTRE TRI. Research Report 6/2005 of The Institute of Systems Engineering and Computers (INESC-Coimbra), Coimbra, Portugal. Available from: <http://www.inescc.pt>.